AMENDMENTS

In the Claims

Please amend the claims as indicated hereafter. [Use ~~strikethrough~~ for deleted matter and underlined for added matter.]

1. (Currently amended) A method for optimizing compilation time of a program, the program including at least one block of code, said method comprising steps of:

generating a current hash value for a block of intermediate code in the program wherein the prior hash value is associated with preexisting object code that corresponds to the block of intermediate code;

~~skipping optimization of the block of code if the current hash value equals a prior hash value; and~~

~~storing the current hash value in the block of code if the hash value is not equal to the prior hash value for the block of code~~

generating a current object file from the block of intermediate code when the current hash value for the block of intermediate code does not correspond to the prior hash value;

linking the generated current object file with other blocks of object code associated with the program;

retrieving the preexisting object code when the current hash value for the block of intermediate code corresponds to the prior hash value; and

linking the preexisting object code with other blocks of object code associated with the program.

2. (Currently amended) The method of claim [[1]] 28, wherein the storing a hash value step further comprises:

allocating area for the generated hash value.

3. (Original) The method of claim 1, further comprising:

setting a scope of the least one block of code.

4. (Currently amended) The method of claim [[1]] 28, wherein the generating a hash value step further comprises:

using a parameter in hashing function to generate the hash value, wherein the parameter is selected from at least one of the group of a code stream, and a data stream.

5. (Original) The method of claim 1, further comprising the step of:

generating a notice when the hash value is not equal to a prior hash value for the block of code.

6. (Currently amended) A system for optimizing compilation time of a program, the program including at least one block of code, comprising:

means for generating a hash value for a block of code in the program;

~~means for storing the hash value with the block of code if the hash value is not equal to a prior hash value for the block of code; and~~

~~means for skipping optimization of the block of code if the hash value equals the prior hash value~~

means for generating a current object file from the block of intermediate code when the current hash value for the block of intermediate code does not correspond to the prior hash value;

means for linking the generated current object file with other blocks of object code associated with the program;

means for retrieving the preexisting object code when the current hash value for the block of intermediate code corresponds to the prior hash value; and

means for linking the preexisting object code with other blocks of object code associated with the program.

7. (Original) The system of claim [[6]] 29, wherein the storing means further comprises:

means for allocating area for the generated hash value.

8. (Original) The system of claim 6, further comprising:

means for setting a scope of the least one block of code.

9. (Currently amended) The system of claim [[6]] 29, wherein the hash value is generated using a parameter in the block of code, wherein the parameter is selected from at least one of the group of a code stream, and a data stream.

10. (Original) The system of claim 6, further comprising:

means for generating a notice when the hash value is not equal to a prior hash value for the block of code.

11. (Currently amended) A computer readable medium for optimizing compilation time of a program, the program including at least one block of code, comprising:

logic for generating a hash value for a block of code in the program;

~~logic for storing the hash value with the block of code if the hash value is not equal to a prior hash value for the block of code; and~~

~~logic for skipping optimization of the block of code if the hash value equals the prior hash value~~

logic for generating a current object file from the block of intermediate code when the current hash value for the block of intermediate code does not correspond to the prior hash value;

logic for linking the generated current object file with other blocks of object code associated with the program;

logic for retrieving the preexisting object code when the current hash value for the block of intermediate code corresponds to the prior hash value; and

logic for linking the preexisting object code with other blocks of object code associated with the program.

12. (Currently amended) The computer readable medium of claim [[11]] 30, wherein said logic for storing a hash value further comprises:

logic for allocating area for the generated hash value.

13. (Original) The computer readable medium of claim 11, further comprising:

logic for setting a scope of the least one block of code.

14. (Currently amended) The computer readable medium of claim [11] 30, wherein the hash value is generated using a parameter in the block of code, wherein the parameter is selected from at least one of the group of a code stream, and a data stream.

15. (Original) The computer readable medium of claim 11, further comprising:

logic for generating a notice when the hash value is not equal to a prior hash value for the block of code.

16. (Currently amended) A system for optimizing compilation time of a program, comprising:

a compiler that generates at least one block of code from the program; and

a compilation optimizer, wherein the compilation optimizer further comprises:

logic that generates a hash value for a block of code in the program;

~~logic that stores the hash value with the block of code if the hash value is not equal to a prior hash value for the block of code; and~~

~~logic that skips optimization of the block of code if the hash value equals the prior hash value~~

logic that generates a current object file from the block of intermediate code when the current hash value for the block of intermediate code does not correspond to the prior hash value;

logic that links the generated current object file with other blocks of object code associated with the program;

logic that retrieves the preexisting object code when the current hash value for the block of intermediate code corresponds to the prior hash value; and

logic that links the preexisting object code with other blocks of object code associated with the program.

17. (Currently amended) The system of claim [[16]] 31, wherein the compilation optimizer further comprises:

logic that allocates area for the generated hash value.

18. (Original) The system of claim 16, wherein the compilation optimizer further comprises:

logic that sets a scope of the least one block of code.

19. (Currently amended) The system of claim [[16]] 31, wherein the hash value is generated using a parameter in the block of code, wherein the parameter is selected from at least one of the group of a code stream, and a data stream.

20. (Original) The system of claim 16, wherein the compilation optimizer further comprises:

logic that generates a notice when the hash value is not equal to a prior hash value for the block of code.

21. – 24. (Canceled)

25. (Previously presented) A method for compiling a program, comprising:

generating a current hash value for a block of intermediate code in the program;

comparing the current hash value with a prior hash value associated with preexisting object code that corresponds to the block of intermediate code;

linking the preexisting object code with other blocks of object code associated with the program when the current hash value corresponds to the prior hash value; and

when the current hash value for the block of intermediate code does not correspond to the prior hash value,

generating a current object file from the block of intermediate code when the current hash value for the block of intermediate code does not correspond to the prior hash value; and

linking the generated current object file with other blocks of object code associated with the program.

26. (Previously presented) The method of claim 25, further comprising changing a portion of the block of intermediate code in the program such that generating the current hash value and such that comparing the current hash value with the prior hash value validates software maintenance changes.

27. (Previously presented) The method of claim 25, further comprising changing a portion of the block of intermediate code in the program such that generating the current hash value and such that comparing the current hash value with the prior hash value identifies changes in the program.

28. (New) The method of claim 1, further comprising:

skipping optimization of the block of code if the current hash value equals a prior hash value; and

storing the current hash value in the block of code if the hash value is not equal to the prior hash value for the block of code.

29. (New) The system of claim 6, further comprising:

means for storing the hash value with the block of code if the hash value is not equal to a prior hash value for the block of code; and

means for skipping optimization of the block of code if the hash value equals the prior hash value.

30. (New) The computer readable medium of claim 11, further comprising:

logic for storing the hash value with the block of code if the hash value is not equal to a prior hash value for the block of code; and

logic for skipping optimization of the block of code if the hash value equals the prior hash value.

31. (New) The system of claim 16, further comprising:

logic that stores the hash value with the block of code if the hash value is not equal to a prior hash value for the block of code; and

logic that skips optimization of the block of code if the hash value equals the prior hash value.